Adventures in Porting Rational Apex to Linux

Presentation to ACM SIGAda 2003
Rational Apex product line

Apex Ada and Duo (Ada + C/C++)
Apex Embedded
TestMate
AXI, etc.

has been repackaged …

IBM Rational Ada Developer
IBM Rational Ada Developer

- Internally, product name remains “Apex”
- Apex 4.2.2
- Supported distributions

<table>
<thead>
<tr>
<th>Red Hat 8.0</th>
<th>SuSE SLES 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red Hat 9.0</td>
<td>Red Hat Enterprise 3</td>
</tr>
<tr>
<td>SuSE 8.2</td>
<td>SuSE 9.0 (soon)</td>
</tr>
</tbody>
</table>
Why Linux?

- Open source/open standards ideal
- Increasing perception as a real OS
- Rapid growth
- Cost of ownership

Specifically, for Apex

- Numerous customer inquiries
- Several specific requests
- IBM backing for Linux
The real reason (speaking as an engineer)

- Linux is cool
Development Game Plan

Leverage existing components

- X86 compiler (Apex Embedded)
- Pentium II/III optimizations (Apex for Windows)
- POSIX-based runtimes (UNIX platforms)
- Elf OMF generation (Sparc)
- Threaded cross debugger (Apex for LynxOS)
- Experience with recent embedded ports
Development Goals

- X86 – specifically, Pentium
- Red Hat 7.3 and 8.0
- Threaded runtime
Challenges
Dynamic vs. Static Libraries

- Static linking is easier on UNIXes – harder on Linux
- Threading is broken in static system libraries
- Cannot mix-and-match system libraries
- Big gap between Red Hat 7.3 and 8.0
- New POSIX in Red Hat 9.0 is broken
  *Apex works around this*
Ada Priorities

- Ada task = Linux thread
- Linux: 3 scheduling policies
  - Round-Robin (RR), FIFO, “other”
  - RR and FIFO have priorities
  - “other” only 1 priority value – i.e., no priorities
- Punch line: RR and FIFO require privileges
- Usability dilemma
Signals

- Under Linux, every thread is a process
- Signal handling
- Note a change is in the works – Native POSIX Thread Library (inc. in RH 9.0)
Stack Limit Check

- Stack Limit checking required for Ada
- Most CPUs: dedicate a register
- Not feasible on x86
- Runtime call
  - Rational Exec – not too burdensome
  - Linux – potential OS call, too burdensome
NFS

- Solaris client, Linux host – slow!
- Investigation
  - Linux server: NFS over TCP or UDP
  - Linux client: UDP
  - Common automount map => all clients use same
  - Result: Sol->Linux slow, or Linux->Linux broken
- Fix: Patch to nfsmount.c
Conclusions
Conclusions

- **Linux:**
  Definitely usable, but still a bit of a work-in-progress
- **Don’t bother with static libraries**
- **Fundamental limitation wrt task priorities**